MERGING CONCURRENT BEHAVIORS ON A REDUNDANT MANIPULATOR

Paul G. Backes and Mark K. Long
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California, 91109

Abstract

Task space control of manipulators is extended both in the number of simultaneous behaviors and in the dimension of the task space. An application is decomposed into multiple simultaneous behaviors whose resultant behavior will provide the motion necessary to execute the task. Each behavior generates commands in its own coordinate system. These simultaneous commands are merged in a motion space using impedance control to compute a resultant command to the manipulator. The task space of each behavior can have the dimensionality of the mechanism being controlled. Control of a seven degree of freedom manipulator is described here so the available task space for each behavior has dimensionality seven.

1 Introduction

A large number of motion sources may be necessary in a robot control system which is expected to perform a wide variety of tasks. For example, a trajectory generator may be needed to provide position setpoints, force sensor inputs may be necessary for contact applications, hand controller inputs may be needed for operator control, gripper motion for grasps, and visual feedback for automatic alignment. Many tasks require a simultaneous combination of motion sources. A compliant grasp task requires simultaneous force control and gripper control. A shared control polishing task could use hand controller inputs to specify motion tangential to a surface while force control controlled the force of contact with the surface and the manipulator reconfigured itself in real time to stay away from joint limits, singularities and collisions.

The necessary number of degrees of freedom (DOFs) of the mechanism can also vary depending on the task. A four DOF scara manipulator is sufficient for many pick and place operations. A six DOF manipulator is sufficient for placing objects in an arbitrary orientation. A seven DOF manipulator provides an ability to continuously change its internal link configuration for a constant tool position and orientation, and can extend the dextrous workspace. For the seven DOF manipulator, the possible dimension of the output motion (seven) is greater than the dimension of possible motion of the gripper (six). All of the mechanism DOFs should be available for task execution. The control scheme must therefore allow both a variable number of simultaneous input sources and a variable dimension task space.

There are different ways to implement a system to provide multiple input sources. One solution is to provide a flexible robot programming environment which provides a layered set of subroutines for robot applications programming. A custom program could then be developed to utilize the needed sensors for a specific task. A robot language could also be used to develop a program to merge control based upon multiple sensors. The approach used to implement the control architecture of this paper is to provide a fixed software system with data driven execution. The control system provides a large suite of capabilities based upon input data. This approach is used to satisfy requirements for space telerobotics where the flight component of the telerobotic system must be flight qualified. The fixed flight software can provide the multiple control sources with the behavior of control from each source dependent on the parameterization data which can be sent from a distant ground station. The actual task execution motion is the resultant behavior of all the input sources.

This paper describes a control architecture which allows execution of a task to be considered as the re-

sultant behavior of execution of multiple concurrent behaviors. The dimensionality of the execution space of each behavior and the resultant behavior can be extended to the dimensionality of the controlled mechanism. Task description consists of decomposing the desired execution into the multiple simultaneous behaviors. Each behavior generates motion commands in its own behavior space. These simultaneous commands are merged in a common motion space to compute a resultant command to the manipulator. The task space of each behavior can have the dimensionality of the mechanism being controlled. The architecture is applied to control of a seven DOF manipulator. The results are also applicable to other redundant and non-redundant manipulators with various numbers of 1) 0 1's. Previous work has described techniques for compliant motion control [1, 2, 3], shared control [4], and redundancy resolution [5, 6, 7]. These capabilities become a subset of the more general architecture described here.

2 Control Architecture for Multiple Simultaneous Behaviors

The control architecture for simultaneous execution of multiple control behaviors is shown in Figure 1. The *Application Space* includes all potential application tasks which the robot control system must be able to accomplish. These application tasks could be sequenced together to accomplish a larger task. Execution of a given application task can be decomposed into concurrently executing behaviors. For example, a door opening task could utilize a trajectory generator to generate the nominal trajectory while force control adds small perturbations to adjust for errors between the planned trajectory and the physical system motion. The *Command To Behavior Map* performs the mapping between the task and the required concurrent behaviors. This could be done automatically or through interaction with an operator.

2.1 Behavior Space

The *Behavior Space* includes all of the independent control behaviors. Each control behavior executes in its own control space before the resultant control inputs are transformed to a common motion space. Trajectory Tracking is a control behavior which provides a trajectory generator to generate real-time trajectories. The Teleoperation behavior takes real-time operator inputs and generates control inputs. Dither generates small periodic dither control inputs. Force Tracking provides control of contact forces between the manipulator and the environment. Manipulability computes an optimum arm configuration and generates control inputs to move toward it. Singularity Avoidance generates control inputs to keep the arm away from singularities. Joint Limit Avoidance generates control inputs to keep the arm away from joint limits. Obstacle Avoidance generates control inputs to prevent collisions. Proximity generates control inputs to control proximity to a real or virtual object. Visual Tracking generates control inputs to provide visual servoing. Other behaviors could also be provided. Each behavior has command parameters that specify its operation and use of real and virtual sensor data. Virtual sensors are those that derive data, possibly from real sensors, e.g., a joint limit or singularity sensor derives data from real joint position sensors.

More complex resultant behaviors can be generated by concurrent execution of individual behaviors. For example, a polishing behavior may be composed of teleoperation, force tracking, manipulability, joint limit avoidance, obstacle avoidance, and singularity avoidance behaviors. Teleoperation could allow motion inputs by an operator only tangential to the surface normal. Force tracking could provide a constant force against the surface. Manipulability could control the arm configuration for optimal control of fine forces. Joint limit avoidance, obstacle avoidance and singularity avoidance would keep the arm from collisions and singularities. The operator would then only have to provide the motion over the surface. The autonomous system would provide the rest of the control.

2.2 Motion Space

The *Motion Space* is the common control space for all behaviors. Most general purpose six DOF manipulators will have a motion space defined to be the position and orientation of a tool held in its end-effector. Mechanisms with more than six mechanical DOFs have been referred to as kinematically redundant [8, 9] since the classical problem of end-effector position and orientation control for a spatial manipulator can be handled by a six DOF robot. Task requirements often dictate a task space of dimension greater than six. For so called kinematically redundant robots, a motion space is defined that spans all

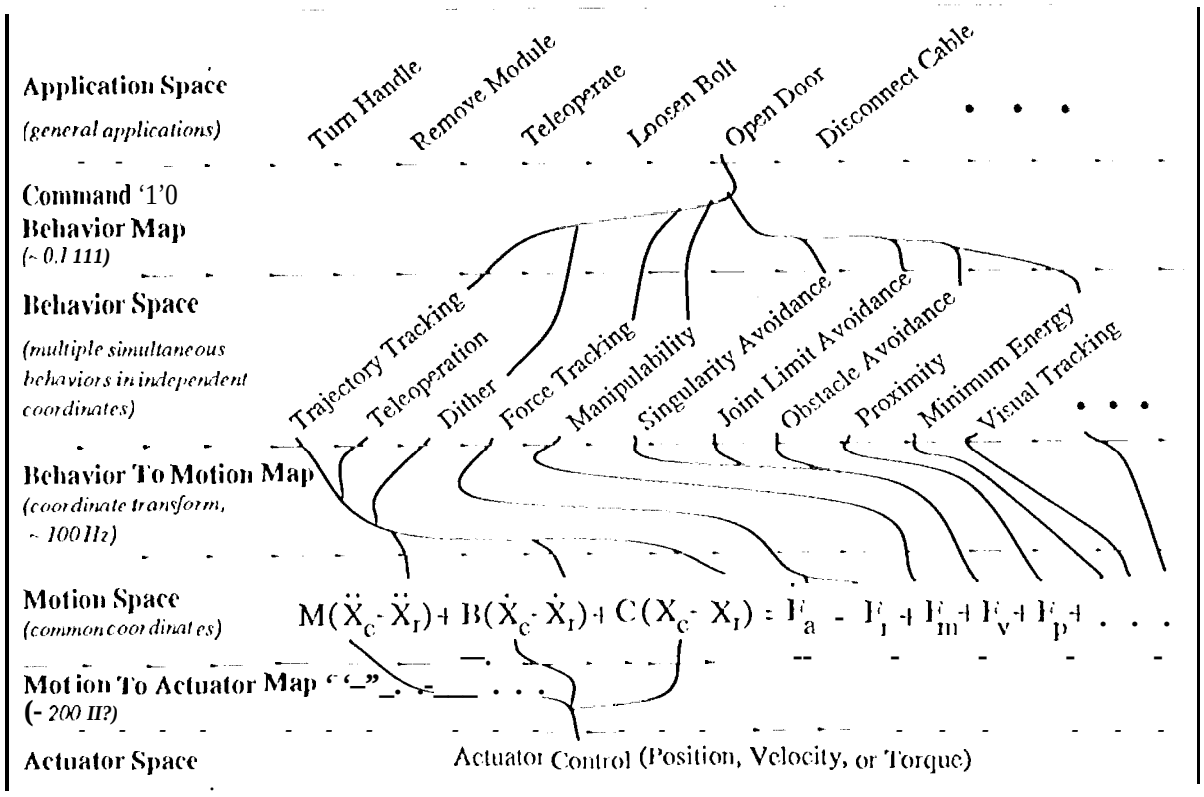

Figure 1: Task decomposition and control for concurrent behaviors

of the mechanical DOFs. The motion space of the seven DOF manipulator used here includes a six DOF coordinate frame (the IMPEDANCE frame), and an "arm angle" parameter which describes the internal configuration of the arm [8, 6]. The arm angle, represented by ψ, is defined as the angle between the plane passing through the shoulder, elbow, and wrist points and some reference plane; we chose the vertical plane here.

Each behavior may compute control inputs in a unique frame but these inputs are transformed into the common motion space to be merged together resulting in a single set of motion commands. Thus each motion DOF can receive inputs from multiple behaviors. Motion Space control is done here using impedance control [1] but with the expanded ability to merge multiple control inputs either as position inputs or force inputs [10]. In addition, the impedance equation can be extended beyond six DOFs to match the dimension of the motion space. The Motion Space impedance control equation, as shown in figure 1, is

$$M \cdot (\ddot{X}_c - \ddot{X}_r) + B \cdot (\dot{X}_c - \dot{X}_r) + K \cdot (X_c - X_r) = \sum F_i \quad (1)$$

where M is the inertia matrix, B is the damping matrix, K is the stiffness matrix, X_r is the reference trajectory, X_c is the commanded position, and $\sum F_i$ is the sum of all behavior inputs mapped to forces. Behaviors can also generate position commands and merged with the reference trajectory. Equation 1 is implemented with

$$^I\ddot{X}_c^{n+1} = {}^I\ddot{X}_r^{n+1} + M^{-1} \cdot [\sum {}^I F_i^n - B \cdot ({}^I\dot{X}_c^n - {}^I\dot{X}_r^{n+1}) - K \cdot ({}^I X_c^n - {}^I X_r^{n+1})] \quad (2)$$

This gives the desired acceleration of the mechanism in the Motion Space. The motion commands are then mapped into the actuator space of the mechanism.

2.3 Actuator Space

The *Actuator Space* is defined as the space of active actuation of the mechanism. Mechanisms which have more than six actuator DOFs fall into two general categories, kinematically redundant and actuationally redundant. Typically, kinematically redundant mechanisms have additional behaviors associated with position and actuationally redundant manipulators have

additional behaviors associated with force [5]. Mapping from the application defined motion space to the actuator space of the mechanism is often referred to as inverse kinematics for position servoed actuators and force to torque map for force-torque servoed actuators. For most applications the motion space should completely span the actuator space of the manipulator to provide the widest array of behaviors for task execution. The mapping is then one-to one and common Jacobian transpose and Jacobian inverse techniques apply [11, 12]. If there are more DOFs in the actuator space than in the motion space, the mapping is underconstrained and a variety of techniques can be used including pseudo-inverse [13] or minimum kinetic energy [5]. Conversely, if there are fewer DOFs in the actuator space than the motion space, the problem is overconstrained and damped least-squares [14, 15, 16] and other techniques are available. Care must be taken to assure that a one-to one mapping between motion space and actuator space does not degenerate at or near a singularity.

The present implementation utilizes joint position servos so inverse kinematics is used to compute joint angle setpoints from the Motion Space acceleration output. The Motion Space velocity vector is computed with

$$ {}^{I}\dot{X}_c^{n+1} = {}^{I}\dot{X}_c^{n} + {}^{I}\ddot{X}_c^{n+1}\Delta_T \qquad (3) $$

Although Jacobian inverse routines could be used, a damped-least-squares inverse is used here to allow further task prioritization and singularity robustness. The damped-least squares inverse is developed in [14, 15, 16] and applied to redundant arms in [6, 17, 7]. The Motion Space velocity vector of the manipulator has three translational coordinates, three orientation coordinates and the arm angle [6]. A composite Jacobian is formed from the individual Jacobians that relate the rate of change of the joint angles to the rate of change of the motion space parameters. Here the composite Jacobian, J^C, is given by [7]:

$$ J^C_{7\times 7} = \begin{pmatrix} J^{\omega}_{3\times 7} \\ J^{v}_{3\times 7} \\ J^{\psi}_{1\times 7} \end{pmatrix} \qquad (4) $$

where J^{ω} is the angular velocity Jacobian, J^v is the linear velocity Jacobian, and J^{ψ} is the arm angle Jacobian. J^{ω} and J^v are readily available using [18]:

$$ \begin{pmatrix} J^{\omega}_r \\ J^{v}_r \end{pmatrix} = \begin{pmatrix} \hat{z}_1 & \hat{z}_2 & \cdots & \hat{z}_7 \\ \hat{z}_1 \times \mathbf{P}_{1,r} & \hat{z}_2 \times \mathbf{P}_{2,r} & \cdots & \hat{z}_7 \times \mathbf{P}_{7,r} \end{pmatrix} \qquad (5) $$

where r is the velocity reference point, $\hat{z}_i$ is the z axis of joint i, and $\mathbf{P}_{i,r}$ is the position vector from the ith link frame to the velocity reference point r [6]. The arm angle Jacobian is available from [6, 7]:

$$ J^{\psi} = \frac{(\hat{\mathbf{w}}\times\mathbf{p})^T}{\|\mathbf{p}\|^2}\mathbf{E} + \left\{ \frac{\dot{\mathbf{V}}_T^T\mathbf{w}}{\|\mathbf{h}\|^2}(\hat{\mathbf{w}}\times\mathbf{h})^T - \frac{\dot{\mathbf{w}}^T\mathbf{e}}{\|\mathbf{w}\|\|\mathbf{p}\|^2}(\hat{\mathbf{w}}\times\mathbf{p})^T \right\}\mathbf{W} \qquad (6) $$

where w : $\mathbf{P}_{0,7}$, e : 1'0,4 , p : e - $\hat{\mathbf{w}}(\hat{\mathbf{w}}^T\mathbf{e})$, $\mathbf{V}_T$ is the vector specifying the reference plane, II : (w × $\hat{\mathbf{V}}$) x w , II; is the elbow linear velocity Jacobian, arid $\mathbf{W}$ is the wrist linear velocity Jacobian. Notice that most of the required data for J^C is available as a by product from a forward kinematic iteration [7].

Now with the motion space command vector, $\dot{X}_c$, and the motion to actuator space map, J^C, the joint servo velocity commands can be computed using dalll~)ecl-least squares with

$$ \dot{\theta}_d = [J^T \cdot W_T \cdot J + W_V]^{-1} \cdot J^T \cdot W_T \cdot \dot{X}_c \qquad (7) $$

where W_T is a diagonal task weighting matrix that relates the relative priorities of the tasks. W_V is a diagonal velocity weighting matrix which weights the norm of joint velocities. It is important to note that while a non-zero W_V matrix will provide robustness to singularities by limiting excess joint velocities, it will also induce tracking error over the entire workspace. By setting W_T to identity and W_V to zero a standard inverse Jacobian result is provided with the same algorithm.

3 Individual Behaviors

As shown in figure 1, various individual behaviors can execute concurrently. A behavior can generate either position commands, which are merged with the reference position trajectory on the left side of Equation 1, or force commands which are merged with $\sum F_i$ 011 the right hand side of the equation.

Teleoperation is shown rrs a position based input in Figure 1 but is implemented as a force based input, related to input velocities, here. The input velocity motion by tile operator with a six DOF hand controller is transformed to equivalent velocities, ${}^{T}\dot{X}_h$, in the teleoperation behavior frame, TELEOP(J'), based upon the selected teleoperation mode [10]. These velocities are multiplied by a damping matrix, B_t, to

compute the forces in the TELEOP frame and then transformed to equivalent forces in the IMPEDACE frame with

$$^{I}F_t = {}^{I}_{T}T_f \cdot B_t \cdot {}^{T}\dot{X}_h \qquad (8)$$

where T_f is a rigid body force transformation. The damping matrix, B_t, can be Used to select operator input directions. The operator inputs arm angle velocity by pressing a trigger on the hand controller and changes the sign by pressing a button on the hand controller. All of the above transformations are seven dimensional so the input arm angle is also transformed to a force in the Motion Space.

Forces are not controlled directly with impedance control. Rather, a position set point is specified inside an object and the actual steady state applied force is a function of both the target stiffness and the position error. This approach is available with this implementation, but an alternative approach has also been implemented. In the alternative approach, a reference (desired) force is specified and the difference between the reference and actual forces in the FORCE frame is used on the right side of Equation 1. Then exact force control is possible by setting the reference stiffness, K, and the reference trajectory velocity and acceleration to zero in force controlled DOFs. Assuming that the environment can be modeled as a stiffness with spring constant k_{env}, the applied force in a DOF will be

$$f_a = k_{env}(x_r - x_e) \qquad (9)$$

where x_r is the position at the initial contact point. If $x_e = x_e - x_r$, then the impedance equation (with no stiffness) for this DOF is

$$m\ddot{x}_e + b\dot{x}_e = -k_{env}x_e - f_r \qquad (10)$$

In steady state the desired and actual applied forces will be equal for any target impedance parameters which provide stable contact for the characteristic equation

$$ms^2 + bs + k_{env} = 0 \qquad (11)$$

where m is the mass term of M, b is the damping term of B, and k_{env} is the stiffness in the force controlled DOF. Either approach to control of forces is available if the difference between the reference and actual forces is added to the right hand side of Equation 1. This is shown in figure 1 with the difference $F_a - F_r$.

The trajectory generator behavior computes the reference trajectory acceleration $\ddot{X}_r$, velocity $\dot{X}_r$, and position X_r. This trajectory is computed in the nominal motion frame and then transformed to an equivalent motion in the IMPEDANCE frame. By having a nominal motion frame different from the IMPEDANCE frame, a one degree of freedom rotation in the nominal motion frame is an arc. in the IMPEDANCE frame. This is useful if you have only a linear trajectory generator, as is implemented here, but want to move in arcs such as for door opening. The arm angle is also generated as part of the trajectory.

Joint travel limiting provides an artificial potential field at the end of travel limits on each joint. This field is then mapped to the motion space to resist operator commands that exceed joint limits. While the local site path planner can predict and avoid joint limits in its commands, often the operator cannot. The joint travel limiting sensor resists this motion so the operator does not induce a fault condition which would interrupt the current task. Similar to the joint travel limiting behavior is the behavior which limits motion in the manipulator workspace singular regions. Information in joint space or motion space about the singular regions is required. For the Robotics Research K-1207 arm motion space information about the location of singularities is used from [19]. Some singular regions are qualitatively located at joint limits; these are taken care of by the above behavior. Others are located at configurations when the seventh joint frame is within 0.2 meters or beyond 1.1 meters of the first joint frame. Thus if $\|^{0}P_7\| > 1.1$ meters then

$$F_{singularity} = -(^{motion}_{7}T_f) \cdot K_{singularity} \cdot (\|P_{actual}\| - 1.1) \cdot \hat{P}_{actual} \qquad (12)$$

or if $\|^{0}P_7\| < 0.2$ meters then

$$F_{singularity} = -(^{motion}_{7}T_f) \cdot K_{singularity} \cdot (0.2 - \|P_{actual}\|) \cdot \hat{P}_{actual} \qquad (13)$$

where $K_{singularity}$ is the gain for the singular region avoidance, $^{motion}_{7}T_f$ is the rigid body transformation between the joint 7 frame and the motion space coordinate system. P_{actual} is the actual current position of the joint seven frame, $\hat{P}_{actual}$ is unit vector in the direction of P_{actual}, and $F_{singularity}$ is the singular region avoidance behavior command in the motion space. Note that if the manipulator is not near a singular region there is no commanded motion from this behavior.

Other behaviors can also be added either as position inputs or force inputs. The joint travel limit, trajectory generator, force control, and teleoperation

behaviors have been implemented. Others will be implemented in the coming year.

4 Bounded Behavior Execution

The control scheme for concurrent behaviors merging has been developed for space flight applications. Therefore it as been implemented with a fixed software system as described below in Section 6. An additional feature which is necessary for execution of tasks in a remote space environment is bounded behavior control execution. The multiple concurrent behaviors are merged together to generate the resultant behavior. This resultant behavior must then be monitored during execution to make sure that it stays within specified bounds for safety. The local site can plan tasks and simulate the execution on a local simulator, but cannot be sure of the motion generated by real-time sensor based motion. To ensure safety the local site can specify and verify safety of tasks which execute within specified bounds. These bounds may include the difference between the reference trajectory and the actual trajectory, force thresholds, and proximity thresholds. As long as the execution progresses within the specified bounds, it should be safe.

5 Resultant Behavior Parameterization

The multiple concurrent behaviors control scheme describe above has been implemented as described below in Section 6. The result is a fixed software system which provides a wide range of behaviors for task execution as specified by command parameters. Example parameterization is given here for various tasks. Safety monitoring is used for all tasks with parameterization given in the command interface.

Guarded motion is non-contact motion to a goal point with automatic stop upon sensed collision. A sensed collision implies that, the forces went out of the acceptable bound on the force behavior. The command parameterization specifics a goal point and a time to reach there and the impedance parameters, M, B, and K. Also, the force thresholds are specified.

A Move To Touch behavior is executed as a sequence of two individual behaviors. The first behavior is Guarded Motion to a point inside of the surface to be touched. Collision with the object is sensed by a force monitor and the motion is stopped. The next command again uses the M, B, and K impedance parameters and trajectory generator for motion. The arm moves back toward where the previous command started until the contact force magnitude is less than a given threshold, above a safety threshold, or the arm has returned to that initial point. Monitors for low and high force thresholds are used as well as testing for termination of the trajectory.

For shared control polishing, teleoperation inputs are used instead of the trajectory generator. The TELEOP frame is specified to be at the center of a flat polishing tool held by the manipulator. The teleoperation damping matrix, B_t, is specified so that the operator can input motion only tangential to the polishing tool surface and about the tool surface normal (the diagonal matrix has zeros for these DOFs). Impedance parameters M and B are specified but the stiffness K is set to zero to allow direct force control. The reference forces are set to zero except for along the Z axis of the FORCE frame (which has its X-Y plane the same as the tool surface) which has a negative setpoint so that a constant force will be applied to the surface. If the surface is curved, the contact point may not be the center of the FORCE frame resulting in a generated measured actual torque which will result in a rotation to eliminate the torque. The cleaning tool will therefore maintain its center at the contact point with the surface as the operator moves the tool tangential to the surface.

The door opening behavior can be specified several different ways. The purposely generated motion should be about the hinge axis. This motion can be caused by the trajectory generator, teleoperation input, or force control. The case where the nominal motion is generated with the trajectory generator is described here. The nominal motion frame is placed such that its Z axis is aligned with the hinge axis. A one DOF motion about the Z axis is then specified to result in a reference trajectory that will move the door through the expected arc. The IMPEDANCE frame can be placed coincident with the nominal motion frame or at the grasp point. With the M, B, and K impedance parameters all specified, the motion will follow the desired trajectory with a small resultant error in position which is the balance between the forces generated due to the inaccuracy in the planned tra-

jectory and the stiffness term.

6 Laboratory Implementation

The data driven merging of concurrent behaviors for a redundant manipulator has been developed as part of the Space Station Freedom Advanced Prototype Development Preject for control of Space Station manipulators. The development and implementation has been done in the JPL Supervisory Telerobotics (STELER) laboratory. The STELER lab telerobot system is composed of a local site where task commands are specified by an operator with a graphical interface and a remote site where the commands are executed. The remote site was developed to be able to execute multiple concurrent behaviors as described by local site command parameterization, and has been implemented in Ada to be consistent with language constraints for Space Station systems. The system currently uses a seven DOF Robotics Research Corporation K-1207 dextrous manipulator with a six DOF LORD force-torque sensor at the wrist and a servoed gripper as shown in figure 2, where the control system is used to perform an orbital replacement unit changeout task. Autonomous commands are generated with the local site system and sent for execution at the remote site. For teleoperation and shared control tasks, the operator uses a six DOF hand controller. The system is implemented in a five CPU 68020/68881 environment and generates joint position commands each 2.5 ms which are sent to the manufacturer controller which supplies the joint servo control. Figure 3 shows experimental results of a contact task. The gripper was initially several centimeters above the surface. Direct force control, as explained in Section 3 was used (no trajectory generation (X_r) was used). Only the mass, M (50 in Z), and damping, B (2250 in Z), terms of impedance control were used; the stiffness, K, term was set to zero. The force set point, F_r, was set to -5 N in the IMPEDANCE frame. The force error caused motion in Z to contact and then the force error was reduced to zero resulting in a 5 N stable contact force.

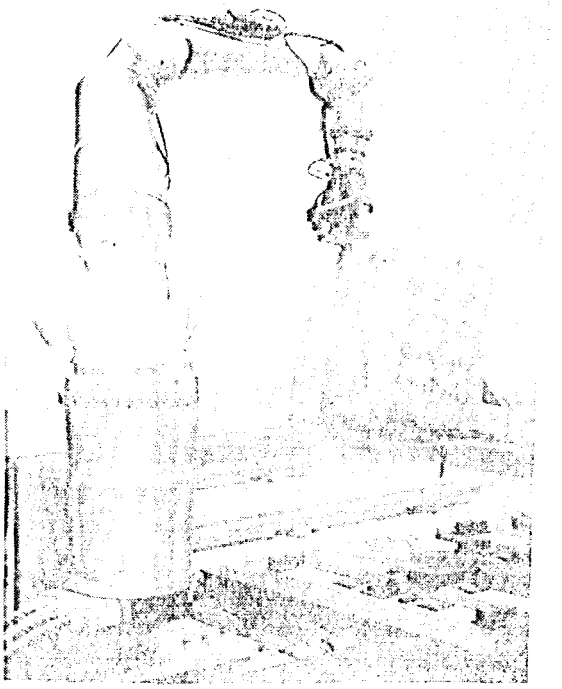

Figure 2: Redundant manipulator performing orbital replacement unit changeout

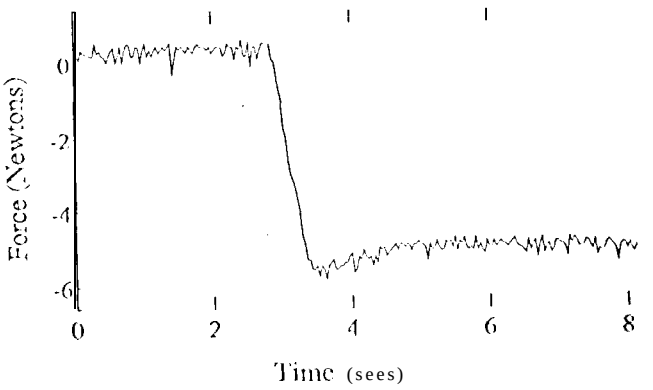

Figure 3: Contact task with 5 N force setpoint

7 Conclusions

A control architecture for data driven merging of concurrent control behaviors has been developed and